LIST OF CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application.

Claims 1-4 (cancelled)

5. (Currently amended) A method for ~~strict priority~~ scheduling ~~in a pile~~ events in a computer processing system, comprising:

~~in a first event in a first node, comprising:~~

identifying multiple queues, each of the multiple queues associated with a corresponding priority;

defining a data structure with a root level having a node group, the node group having k number of nodes, each of the k number of nodes sharing a pointer, each of the k number of nodes stored contiguously in memory, wherein the k number is equal to a number of multiple queues;

associating the multiple queues with respective nodes;

assigning a value representing the corresponding priority to the respective nodes;

~~providing a first queue having a first sort index and a first data field, the first sort index including a first priority and the first data field including a first event queue identifier;~~

~~in a second event in a second node, comprising:~~

~~providing a second queue having a second sort index and a second data field, the second sort index including a first priority and a second data field including a second event queue identifier;~~

determining a priority between the respective nodes ~~first queue and the second queue~~ based on the value ~~first priority in the first queue and the second priority in the second queue~~; and

~~activating~~ selecting one of the multiple queues having a highest priority ~~the first queue if the first priority is a higher priority than the second priority and activating the second queue if the second priority is a higher priority than the first priority~~.

6. (Currently amended) The method of Claim 5, ~~wherein the value of the first priority in the first sort index is equal to the value of the first event queue identifier in the first data field~~ further comprising:

rescheduling the one of the multiple queues after selection.

7. (Currently amended) The method of Claim 6 ~~5~~, wherein the method operation of rescheduling the one of the multiple queues after selection includes,

determining if the one of the multiple queues will be empty after selection;

~~value of the second priority in the second sort index is equal to the value of the second event queue identifier in the second data field~~.

8. (Currently amended) The method of Claim 7 ~~5~~, further comprising:

if the one of the multiple queues will be empty after selection, then the method includes, removing the value representing the corresponding priority from the respective nodes ~~an event from a root node if the first queue in the first event is not empty by rescheduling the event, and percolating a node corresponding to the event down to a location in a heap-like tree structure~~.

9. (Currently amended) The method of Claim 7 ~~5~~, further comprising:

if the one of the multiple queues will not be empty after selection, then the method includes,

retaining the value representing the corresponding priority from the respective nodes, thereby enabling rescheduling of the one of the multiple queues after selection ~~removing an event from a root node if the first queue in the first event is empty by removing the first priority of the first event in the first node; and leaving an empty node in the first node to percolate down a heap-like structure~~.

10. (Currently amended) The method of Claim 5, further comprising:

resolving conflicts between respective nodes assigned a same value by rotating a pointer among the respective nodes assigned the same value ~~inserting an event in the pile by assigning a designated priority corresponding to a particular queue, and placing the designated priority and a corresponding identifier in a node~~.

Claims 11-21 (cancelled)

22. (new) A computer readable medium having program instructions for scheduling events in a computer processing system, comprising:

program instructions for identifying multiple queues, each of the multiple queues associated with a corresponding priority;

program instructions for defining a data structure with a root level having a node group, the node group having k number of nodes, each of the k number of nodes sharing a pointer, each of the k number of nodes stored contiguously in memory, wherein the k number is equal to a number of multiple queues;

program instructions for associating the multiple queues with respective nodes;

program instructions for assigning a value representing the corresponding priority to the respective nodes;

program instructions for determining a priority between the respective nodes based on the value; and

program instructions for selecting one of the multiple queues having a highest priority.

23. (new) The computer readable medium of Claim 22, further comprising:

program instructions for rescheduling the one of the multiple queues after selection.

24. (new) The computer readable medium of Claim 23, wherein the program instructions for_rescheduling the one of the multiple queues after selection includes,

program instructions for determining if the one of the multiple queues will be empty after selection;

25. (new) The computer readable medium of Claim 24, further comprising:

if the one of the multiple queues will be empty after selection, then the computer readable medium includes,

program instructions for removing the value representing the corresponding priority from the respective nodes.

26. (new) The computer readable medium of Claim 24, further comprising:

if the one of the multiple queues will not be empty after selection, then the computer readable medium includes,

program instructions for retaining the value representing the corresponding priority from the respective nodes, thereby enabling rescheduling of the one of the multiple queues after selection.

27. (new) The computer readable medium of Claim 22, further comprising:

program instructions for resolving conflicts between respective nodes assigned a same value by rotating an additional pointer among the respective nodes assigned the same value.